In this work, we investigate the spread of an infectious disease using the Susceptible-Infected-Recovered (SIR) compart
Introduction Understanding the mechanisms driving the spread of infectious diseases in structured populations remains
Our study specifically examines the stochastic spread of a directly transmitted pathogen over a static Erdős-Rényi (ER
The central research questions addressed include: How does network structure modulate the course of an epidemic? W
This paper is organized as follows: Section II details the methodology, including model specification, network construct
S: Susceptible individuals
I: Infectious individuals
R: Removed (recovered/immune) individuals
Transitions in the model are as follows: $\mathbf{S} - (\mathbf{I}) \rightarrow \mathbf{I}$ (transmission upon contact with an infectious individual, with rate β);
A representative basic reproduction number R_0 of 2.5 is chosen as typical for respiratory agents[0].
Network Specification Population structure is modeled as an ER random graph $G(N, p)$ with $N = 1000$ nodes, and a m
Mean degree: $\langle k \rangle = 10$
Second moment: $\langle k^2 \rangle = 110$
The network is constructed using NetworkX's ER generator, and adjacency is stored as a sparse matrix. The degree dis
Parameterization and Initialization Model transition parameters were set as follows:
Transmission rate: $\beta = R_0 \cdot \gamma / q$, where $q = (\langle k^2 \rangle - \langle k \rangle)/\langle k \rangle = 10$ for this network.
Recovery rate: $\gamma = 0.2$ per day (typical for many respiratory infections)
Thus, $\beta = 2.5 \cdot 0.2/10 = 0.05$
Initial conditions assigned 5% of nodes as infectious, the remainder as susceptible: $\{S : 950, I : 50, R : 0\}$.
Stochastic Simulation Simulations were implemented using the FastGEMF platform, which allows for discrete-state sto
Epidemic outcome metrics included:
Epidemic duration (time until $I = 0$)
Peak infection rate (maximum I/N)
Final epidemic size (total R at $t = \infty$)
Peak time (time when I reaches maximum)
Results (populations, compartment counts) were stored at 1-day intervals for analysis. Results Simulation outputs and anal
Epidemic Dynamics Figure shows time series for the mean counts of susceptible, infectious, and recovered individuals a
[ht] [width=0.48]results-11.png Population dynamics of each compartment S, I, R in the SIR model on the ER network
Key Metric Extraction Table quantifies major epidemic outcome metrics:

[ht]

Metric	Value (mean ± SD)
Epidemic Duration (days)	73 ± 3
Peak Infection Fraction	0.26 ± 0.01
Final Epidemic Size (R/N)	0.84 ± 0.01
Peak Time (days)	28 ± 2

Epidemic outcome metrics from simulation ensemble.

Statistical Observations The observed final epidemic size approaches the deterministic SIR prediction for $R_0 = 2.5$ in a
The simulation metrics yielded values consistent with predicted outcomes for $R_0 > 1$: the final epidemic size was rough
Our results also reinforce the link between network structure and control strategies. In more heterogeneous or modular
The major limitation of our study lies in the assumed network simplicity and static interactions. Real populations usua
*References
99

Pastor-Satorras, R., Castellano, C., Van Mieghem, P., and Vespignani, A. Epidemic processes in complex networks. *Reviews*
Barrat, A., Barthelemy, M., and Vespignani, A. *Dynamical Processes on Complex Networks*. Cambridge University Press, 20
Keeling, M. J., and Eames, K. T. D. Networks and epidemic models. *Journal of the Royal Society Interface*, 2(4): 295-307,
Newman, M. E. J. Networks: An Introduction. Oxford University Press, 2018.
Liu, Y., Gayle, A. A., Wilder-Smith, A., and Rocklöv, J. The reproductive number of COVID-19 is higher compared to SAI

*Appendix *Network Construction Code

```
import networkx as nx
import numpy as np
import scipy.sparse as sparse
N = 1000
k_mean = 10
G = nx.erdos_renyi_graph(N, k_mean/(N-1))
A = nx.to_scipy_sparse_array(G)
# Save for use in simulation
sparse.save_npz('output/network.npz', A)
# Calculate degree stats
all_deg = np.array([d for n, d in G.degree()])
mean_k = all_deg.mean()
second_moment_k = np.mean(all_deg**2)
print('Mean degree:', mean_k, '2nd moment:', second_moment_k)
```

*Simulation Code

```
import fastgemf as fg
import scipy.sparse as sparse
N = 1000
G_csr = sparse.load_npz('output/network.npz')
SIR_schema = (
    fg.ModelSchema('SIR').define_compartment(['S', 'I', 'R'])
        .add_network_layer('contact_layer')
        .add_node_transition(name='recovery', from_state='I', to_state='R', rate='gamma')
        .add_edge_interaction(name='infection', from_state='S', to_state='I', inducer='I', network_laye
)
instance = (
```